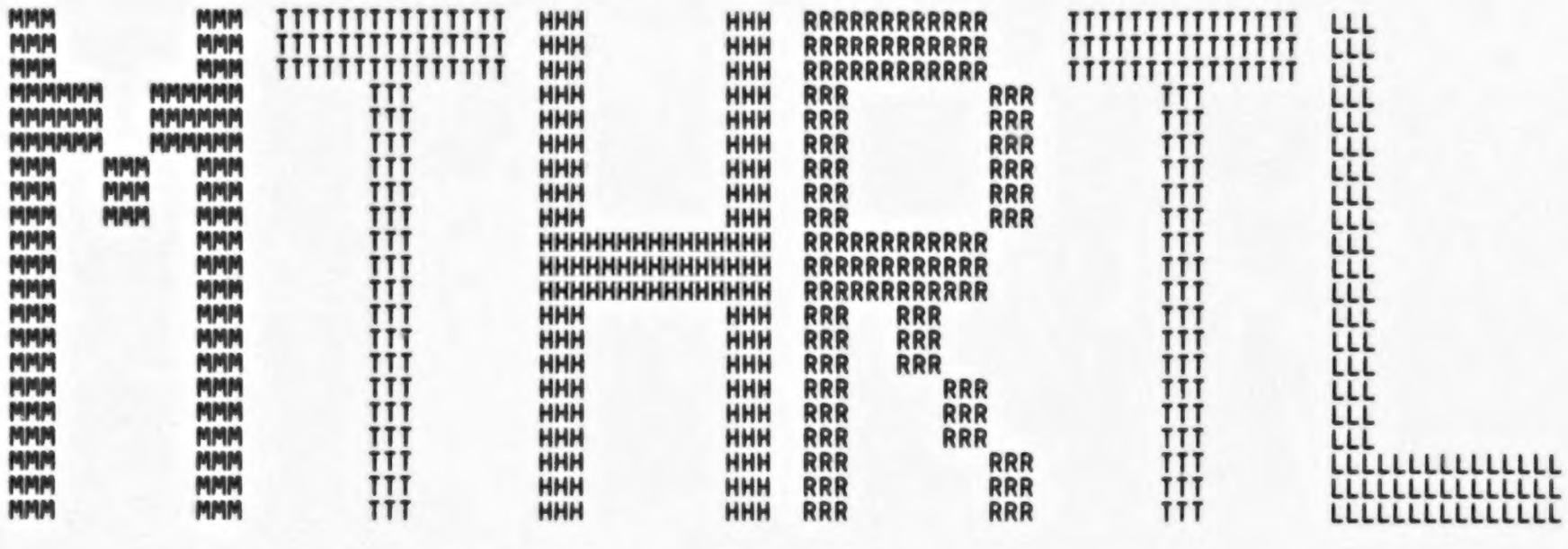
MMM MMM TTTTTTTTTTTTTTT HHH HHH RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
MMMMMM MMMMMM TTT HHH HHH RRR RRR TTT LLL
MMM MMM MMM TTT HHH HHH RRR RRR TTT LLL
MMM MMM MMM TTT HHHHHHHHHHHHHHH RRRRRRRRRRR TTT LLL
MMM MMM MMM TTT HHH HHH RRR RRR TTT LLL
MMM MMM MMM TTT HHH HHH RRR RRR TTT LLL
MMM MMM MMM TTT HHH HHH RRR RRR TTT LLLLLLLLLLLLLLL

FILEID**MTHGATAN

```
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG    AAAAAA    TTTTTTTTTT    AAAAAA    NN      NN
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG    AAAAAA    TTTTTTTTTT    AAAAAA    NN      NN
MMMM  MMMM      TT      HH      HH  GG            AA    AA      TT      AA    AA    NN      NN
MMMM  MMMM      TT      HH      HH  GG            AA    AA      TT      AA    AA    NN      NN
MM  MM  MM      TT      HH      HH  GG            AA    AA      TT      AA    AA    NNNN    NN
MM  MM  MM      TT      HH      HH  GG            AA    AA      TT      AA    AA    NNNN    NN
MM      MM      TT      HHHHHHHHHH  GG            AA    AA      TT      AA    AA    NN  NN  NN
MM      MM      TT      HHHHHHHHHH  GG            AA    AA      TT      AA    AA    NN  NN  NN
MM      MM      TT      HH      HH  GG  GGGGGG    AAAAAAAAAA      TT      AAAAAAAAAA    NN    NNNN
MM      MM      TT      HH      HH  GG  GGGGGG    AAAAAAAAAA      TT      AAAAAAAAAA    NN    NNNN
MM      MM      TT      HH      HH  GG      GG    AA    AA        TT      AA    AA      NN      NN      ....
MM      MM      TT      HH      HH  GG      GG    AA    AA        TT      AA    AA      NN      NN      ....
MM      MM      TT      HH      HH    GGGGGG      AA    AA        TT      AA    AA      NN      NN      ....
MM      MM      TT      HH      HH    GGGGGG      AA    AA        TT      AA    AA      NN      NN      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          .TITLE  MTH$GATAN       ; G Floating Point Arc Tangent Functions
0000     2                                  ; (GATAN,GATAN2,GATAND,GATAND2)
0000     3          .IDENT /2-005/          ; File: MTHGATAN.MAR  EDIT:  RNH2005
0000     4 ;
0000     5 ;**********************************************************************************
0000     6 ;*                                                                                *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                       *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                        *
0000     9 ;*  ALL RIGHTS RESERVED.                                                          *
0000    10 ;*                                                                                *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED         *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER         *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY         *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY        *
0000    16 ;*  TRANSFERRED.                                                                  *
0000    17 ;*                                                                                *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE         *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT        *
0000    20 ;*  CORPORATION.                                                                  *
0000    21 ;*                                                                                *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS         *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                       *
0000    24 ;*                                                                                *
0000    25 ;*                                                                                *
0000    26 ;**********************************************************************************
0000    27 ;
0000    28 ;
0000    29 ; FACILITY: MATH LIBRARY
0000    30 ;++
0000    31 ; ABSTRACT:
0000    32 ;
0000    33 ; MTH$GATAN  is  a  function  which  returns  the G floating point arc-
0000    34 ; tangent value in radians of  its  G floating point argument.
0000    35 ; MTH$GATAN2  is  two  argument G floating arctangent. The call is
0000    36 ; standard call-by-reference.
0000    37 ; MTH$GATAN_R7  is  a  special routine which is the same as MTH$GATAN
0000    38 ; except a  faster non-standard JSB call is used with the argument in
0000    39 ; R0 and no registers are saved.
0000    40 ;
0000    41 ; MTH$GATAND is  a  function  which  returns  the G floating point arc-
0000    42 ; tangent value in radians of  its  G floating point argument.
0000    43 ; MTH$GATAND2  is  two  argument G floating arctangent. The call is
0000    44 ; standard call-by-reference.
0000    45 ; MTH$GATAND_R7  is  a  special routine which is the same as MTH$GATAND
0000    46 ; except a  faster non-standard JSB call is used with the argument in
0000    47 ; R0 and no registers are saved.
0000    48 ;
0000    49 ;--
0000    50 ;
0000    51 ; VERSION: 1
0000    52 ;
0000    53 ; HISTORY:
0000    54 ; AUTHOR:
0000    55 ;       Steven B. Lionel, 15-Jan-79: Version 1
0000    56 ;
0000    57 ; MODIFIED BY:
```

```
0000     58 ;
0000     59 ; VERSION: 2
0000     60 ;
0000     61 ; HISTORY:
0000     62 ; AUTHOR:
0000     63 ;       Bob Hanek, 08-Jun-81: Version 2
0000     64 ;
0000     65 ; MODIFIED BY:
0000     66 ;
0000     67 ;
0000     68 ;
```

```
                     ;
                     ; ALGORITHMIC DIFFERENCES FROM FP-11/C ROUTINE:
                     ;       1. To avoid various flags subroutine calls have been used.
                     ;
                     ; Edit history for Version 1 of MTH$GATAN
                     ;
                     ; 1-001 - Adapted from MTH$DATAN version 1-001.  SBL 15-Jan-79
                     ; 1-002 - Added degree entry points. RNH 15-MAR-1981
                     ;
                     ; Edit history for Version 2 of MTH$GATAN
                     ;
                     ; 2-002 - Use G^ addressing for externals.  SBL 24-August-1981
                     ; 2-003 - Change MTH$$AB_ATAN to MTH$$AB_ATAN_V.  RNH 29-Sep-81
                     ; 2-004 - Change MTH$GATAN2D entry to MTH$GATAND2 in order to conform
                     ;         with original specification.  RNH 05-Oct-81
                     ; 2-005 - Un-did previous edit to conform with PL/1.
                     ;       - Modified small argument logic to avoid a microcode bug in the
                     ;         FPA. RNH 18-Dec-81
```

```
                                        .SBTTL  DECLARATIONS    ; Declarative Part of Module
                                ;
                                ; INCLUDE FILES:          MTHJACKET.MAR, MTHATAN.MAR
                                ;
                                ; EXTERNAL SYMBOLS:
                                ;
                                        .DSABL  GBL
                                        .EXTRN  MTH$K_INVARGMAT
                                        .EXTRN  MTH$$SIGNAL                     ; Signal SEVERE error
                                        .EXTRN  MTH$$AB_ATAN_V                  ; Gobal table used by all Arctangent
                                                                                ;    routines.  Part of MTHATAN.MAR
                                ;
                                ; EQUATED SYMBOLS:
                                ;
                000040FC        ACMASK = ^M<IV, R2, R3, R4, R5, R6, R7> ; .ENTRY register mask, int
                                                                        ; ovf enabled
                                ;
                                ; MACROS:        none
                                ;
                                ; PSECT DECLARATIONS:
                                ;
                00000000        .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                                                                        ; program section for math routines
                                ;
                                ; OWN STORAGE:  none
                                ;
                                ;
                                ; CONSTANTS:
                                ;
                                G_M1.0:
00000000 0000C010               .G_FLOATING     -1.0
                                ;
```

```
                        0008   129 ;
                        0008   130 ;   ***************************** Constants for GATAN ****************************
                        0008   131 ;
                        0008   132 ;
                        0008   133 ;   Each  entry of the GATAN_TABLE contains the the values of XHI, GATAN_XHI_LO
                        0008   134 ;   and GATAN_XHI_HI respectively.  The table is indexed by a pointer  obtained
                        0008   135 ;   from the MTH$$AB_ATAN_V table. The MTH$$AB_ATAN table is common to all of
                        0008   136 ;   the arctangent  routines  and is included as part of the MTHATAN module.
                        0008   137 ;   NOTE:  For performance reasons it is important to have  the  GATAN_TABLE
                        0008   138 ;   longword aligned.
                        0008   139 ;
                        0008   140 ;
                        0008   141          .ALIGN  LONG
                        0008   142
                        0008   143 GATAN_TABLE:
                        0008   144  ;  Entry  0
0000C000 FF0F3FDA       0008   145          .QUAD   ^X0000C000FF0F3FDA      ; 0.10545442998409271E+00
A7116ADD BF29BC67       0010   146          .QUAD   ^XA7116ADDBF29BC67      ; -.25746251625939183E-17
566FD64F E59C3FDA       0018   147          .QUAD   ^X566FD64FE59C3FDA      ; 0.10506611091781236E+00
                        0020   148  ;  Entry  1
00000000 7F6E3FE0       0020   149          .QUAD   ^X000000007F6E3FE0      ; 0.12888884544372559E+00
DF4E8FD5 EFD23C87       0028   150          .QUAD   ^XDF4E8FD5EFD23C87      ; 0.10380935117142230E-16
9B8FE6F7 68453FE0       0030   151          .QUAD   ^X9B8FE6F768453FE0      ; 0.12818216111847078E+00
                        0038   152  ;  Entry  2
0000C000 FEC73FE3       0038   153          .QUAD   ^X0000C000FEC73FE3      ; 0.15621277689933777E+00
78853417 5EC23C83       0040   154          .QUAD   ^X788534175EC23C83      ; 0.84004638358227629E-17
66D21962 D5BE3FE3       0048   155          .QUAD   ^X66D21962D5BE3FE3      ; 0.15496040572616337E+00
                        0050   156  ;  Entry  3
00008000 FC9D3FE8       0050   157          .QUAD   ^X00008000FC9D3FE8      ; 0.19520920515060425E+00
5A36F2BA FC573C87       0058   158          .QUAD   ^X5A36F2BAFC573C87      ; 0.10402146584359310E-16
59EE355B AD2C3FE8       0060   159          .QUAD   ^X59EE355BAD2C3FE8      ; 0.19278481107058049E+00
                        0068   160  ;  Entry  4
00002000 F87A3FEF       0068   161          .QUAD   ^X00002000F87A3FEF      ; 0.24977041780948639E+00
7ED48C95 CA7DBC72       0070   162          .QUAD   ^X7ED48C95CA7DBC72      ; -.40746438836046411E-17
78D54986 54613FEF       0078   163          .QUAD   ^X78D5498654613FEF      ; 0.24476257410146354E+00
                        0080   164  ;  Entry  5
0000E000 FB723FF3       0080   165          .QUAD   ^X0000E000FB723FF3      ; 0.31222221255302429E+00
196AA37E FCD6BC9F       0088   166          .QUAD   ^X196AA37EFCD6BC9F      ; -.27744863807258262E-16
15D7AA33 5E513FF3       0090   167          .QUAD   ^X15D7AA335E513FF3      ; 0.30263177510309219E+00
                        0098   168  ;  Entry  6
0000C000 F3D13FF8       0098   169          .QUAD   ^X0000C000F3D13FF8      ; 0.38988155126571655E+00
1321B0FD 909F3C8D       00A0   170          .QUAD   ^X1321B0FD909F3C8D      ; 0.12821747438427643E-16
1A5C2DEF CACE3FF7       00A8   171          .QUAD   ^X1A5C2DEFCACE3FF7      ; 0.37175325856916230E+00
                        00B0   172  ;  Entry  7
0000C000 E6763FFF       00B0   173          .QUAD   ^X0000C000E6763FFF      ; 0.49844139814376831E+00
AF44EC51 8F0BBC85       00B8   174          .QUAD   ^XAF44EC518F0BBC85      ; -.93496285723919294E-17
6601F613 97F53FFD       00C0   175          .QUAD   ^X6601F61397F53FFD      ; 0.46239995032155440E+00
                        00C8   176  ;  Entry  8
00006000 DF1D4004       00C8   177          .QUAD   ^X00006000DF1D4004      ; 0.65223568677902222E+00
DC171556 EF7DBCA2       00D0   178          .QUAD   ^XDC171556EF7DBCA2      ; -.32847860491614434E-16
96B7173B 7E874002       00D8   179          .QUAD   ^X96B7173B7E874002      ; 0.57794527566576093E+00
                        00E0   180  ;  Entry  9
00004000 C0E2400B       00E0   181          .QUAD   ^X00004000C0E2400B      ; 0.86729538440704346E+00
C4357EF7 E0933CAC       00E8   182          .QUAD   ^XC4357EF7E0933CAC      ; 0.50090445545984688E-16
5CC77669 DCC54006       00F0   183          .QUAD   ^X5CC77669DCC54006      ; 0.71444962622890607E+00
                        00F8   184  ;  Entry 10
00000000 B97B4012       00F8   185          .QUAD   ^X00000000B97B4012      ; 0.11702833175659180E+01
```

```
                DE5AE193 4F4A3CA4  0100   186          .QUAD   ^XDE5AE1934F4A3CA4      ; 0.35231776430069749E-16
                4CCA4044 A36C400B  0108   187          .QUAD   ^X4CCA4044A36C400B      ; 0.86369907905682308E+00
                                   0110   188  ;  Entry 11
                00006000 51BD401A  0110   189          .QUAD   ^X00006000051BD401A      ; 0.16449559926986694E+01
                54459635 8AA53C85  0118   190          .QUAD   ^X54459635 8AA53C85      ; 0.93421751035229863E-17
                203A1861 64A84010  0120   191          .QUAD   ^X203A186164A84010      ; 0.10245743706054911E+01
                                   0128   192  ;  Entry 12
                0000A000 91004024  0128   193          .QUAD   ^X0000A00091004024      ; 0.25708019733428955E+01
                D4B2DF90 D0CBBCB4  0130   194          .QUAD   ^XD4B2DF90D0CBBCB4      ; -.72218657650365456E-16
                88B84B38 32794013  0138   195          .QUAD   ^X88B84B3832794013      ; 0.11998227060617364E+01
                                   0140   196  ;  Entry 13
                0000C000 6FA14035  0140   197          .QUAD   ^X0000C0006FA14035      ; 0.53590154647827148E+01
                61CF645D A10DBCBA  0148   198          .QUAD   ^X61CF645DA10DBCBA      ; -.92388286592139436E-16
                FBF74646 2E5A4016  0150   199          .QUAD   ^XFBF746462E5A4016      ; 0.13863165612417541E+01
                                   0158   200
                                   0158   201  ;
                                   0158   202  ; Tables to be used in POLYG for computing GATAN:  GATANTAB1 is obtained
                                   0158   203  ; from Hart et. al. (No. 4904).  GATANTAB2 is the same as GATANTAB1 except
                                   0158   204  ; that C0 is set to 0
                                   0158   205  ;
                                   0158   206
                                   0158   207  GATANTAB1:
                696A2F61 1F943FD3  0158   208          .QUAD   ^X696A2F611F943FD3      ; C6 = 0.74700604980000002E-01
                974A29A9 43E3BFD7  0160   209          .QUAD   ^X974A29A943E3BFD7      ; C5 = -.90879628821849995E-01
                BAB6DA1C 71C33FDC  0168   210          .QUAD   ^XBAB6DA1C71C33FDC      ; C4 = 0.11111091685300320E+00
                3DD890DF 4924BFE2  0170   211          .QUAD   ^X3DD890DF4924BFE2      ; C3 = -.14285714219884826E+00
                04029999 99993FE9  0178   212          .QUAD   ^X04029999999993FE9      ; C2 = 0.19999999999893708E+00
                554A5555 5555BFF5  0180   213          .QUAD   ^X554A55555555BFF5      ; C1 = -.33333333333333270E+00
                00000000 00004010  0188   214          .QUAD   ^X0000000000004010      ; C0 = 0.10000000000000000E+01
                         00000007  0190   215  GATANLEN1 = .- GATANTAB1/8
                                   0190   216                                  .
                                   0190   217  GATANTAB2:
                696A2F61 1F943FD3  0190   218          .QUAD   ^X696A2F611F943FD3      ; C6 = 0.74700604980000002E-01
                974A29A9 43E3BFD7  0198   219          .QUAD   ^X974A29A943E3BFD7      ; C5 = -.90879628821849995E-01
                BAB6DA1C 71C33FDC  01A0   220          .QUAD   ^XBAB6DA1C71C33FDC      ; C4 = 0.11111091685300320E+00
                3DD890DF 4924BFE2  01A8   221          .QUAD   ^X3DD890DF4924BFE2      ; C3 = -.14285714219884826E+00
                04029999 99993FE9  01B0   222          .QUAD   ^X04029999999993FE9      ; C2 = 0.19999999999893708E+00
                554A5555 5555BFF5  01B8   223          .QUAD   ^X554A55555555BFF5      ; C1 = -.33333333333333270E+00
                00000000 00000000  01C0   224          .QUAD   ^X0000000000000000      ; C0 = 0.00000000000000000E+00
                         00000007  01C8   225  GATANLEN2 = .- GATANTAB2/8
                                   01C8   226
                                   01C8   227  G_PI:
                2D185444 21FB4029  01C8   228          .QUAD   ^X2D18544421FB4029      ; pi
                                   01D0   229  G_PI_OVER_2:
                2D185444 21FB4019  01D0   230          .QUAD   ^X2D18544421FB4019      ; pi/2
                                   01D8   231  G_MPI_OVER_2:
                2D185444 21FBC019  01D8   232          .QUAD   ^X2D18544421FBC019      ; -pi/2
                                   01E0   233  G_PI_OVER_2_HI:
                2D185444 21FB4019  01E0   234          .QUAD   ^X2D18544421FB4019      ; High order bits of pi/2
                                   01E8   235  G_PI_OVER_2_LO:
                5C073314 A6263CB1  01E8   236          .QUAD   ^X5C073314A6263CB1      ; Low order bits of pi/2
                                   01F0   237
```

```
                          01F0   239 ;
                          01F0   240 ; ************************** Constants for ATAND ********************************
                          01F0   241 ;
                          01F0   242 ; Each entry of the GATAND_TABLE contains the the values of XHI, GATAND_XHI_LO
                          01F0   243 ; and GATAND_XHI_HI respectively.  The table is indexed by a pointer  obtained
                          01F0   244 ; from  the MTH$$AB_ATAN_V table. The MTH$$AB_ATAN_V table is common to all
                          01F0   245 ; of the arctangent routines and is included as part of the MTHATAN module.
                          01F0   246 ; NOTE: For performance reasons it is important to have the GATAN_TABLE
                          01F0   247 ; longword aligned.
                          01F0   248 ;
                          01F0   249 ;
                          01F0   250
                          01F0   251 GATAND_TABLE:
                          01F0   252  ;  Entry  0
0000C000 FF0F3FDA         01F0   253          .QUAD   ^X0000C000FF0F3FDA      ; 0.10545442998409271E+00
D413D41C 30D43CDF         01F8   254          .QUAD   ^XD413D41C30D43CDF      ; 0.43285857545785734E-15
1EEC2CFB 14524038         0200   255          .QUAD   ^X1EEC2CFB14524038      ; 0.60198447254440275E+01
                          0208   256  ;  Entry  1
00000000 7F6E3FE0         0208   257          .QUAD   ^X000000007F6E3FE0      ; 0.12888884544372559E+00
C89701C4 A0A03CD8         0210   258          .QUAD   ^XC89701C4A0A03CD8      ; 0.34177440754834374E-15
0F2B59E0 608F403D         0218   259          .QUAD   ^X0F2B59E0608F403D      ; 0.73442968409542955E+01
                          0220   260  ;  Entry  2
0000C000 FEC73FE3         0220   261          .QUAD   ^X0000C000FEC73FE3      ; 0.15621277689933777E+00
8B60C70E 195E3CD0         0228   262          .QUAD   ^X8B60C70E195E3CD0      ; 0.22341992757083616E-15
6BD2E03F C1D44041         0230   263          .QUAD   ^X6BD2E03FC1D44041      ; 0.88785772397440361E+01
                          0238   264  ;  Entry  3
00008000 FC9D3FE8         0238   265          .QUAD   ^X00008000FC9D3FE8      ; 0.19520920515060425E+00
38D734C9 DDBEBCC3         0240   266          .QUAD   ^X38D734C9DDBEBCC3      ; -.13784934004150469E-15
9DCC558C 176D4046         0248   267          .QUAD   ^X9DCC558C176D4046      ; 0.11045756028571212E+02
                          0250   268  ;  Entry  4
00002000 F87A3FEF         0250   269          .QUAD   ^X00002000F87A3FEF      ; 0.24977041780948639E+00
2E402DBD A771BCBA         0258   270          .QUAD   ^X2E402DBDA771BCBA      ; -.92474884414648258E-16
DA96B3EB 0C37404C         0260   271          .QUAD   ^XDA96B3EB0C37404C      ; 0.14023862478771928E+02
                          0268   272  ;  Entry  5
0000E000 FB723FF3         0268   273          .QUAD   ^X0000E000FB723FF3      ; 0.31222221255302429E+00
8E8FD3A0 35F33CEE         0270   274          .QUAD   ^X8E8FD3A035F33CEE      ; 0.83851680459372492E-15
C031026C 56EB4051         0278   275          .QUAD   ^XC031026C56EB4051      ; 0.17339523459959484E+02
                          0280   276  ;  Entry  6
0000C000 F3D13FF8         0280   277          .QUAD   ^X0000C000F3D13FF8      ; 0.38988155126571655E+00
A6895D5D 3CF8BC91         0288   278          .QUAD   ^XA6895D5D3CF8BC91      ; -.14951724532714387E-16
7EE7C536 4CC54055         0290   279          .QUAD   ^X7EE7C5364CC54055      ; 0.21299892736248605E+02
                          0298   280  ;  Entry  7
0000C000 E6763FFF         0298   281          .QUAD   ^X0000C000E6763FFF      ; 0.49844139814376831E+00
BF160F0D 0C67BCFE         02A0   282          .QUAD   ^XBF160F0D0C67BCFE      ; -.16680239163537724E-14
826750B0 7E5A405A         02A8   283          .QUAD   ^X826750B07E5A405A      ; 0.26493565600484001E+02
                          02B0   284  ;  Entry  8
00006000 DF1D4004         02B0   285          .QUAD   ^X00006000DF1D4004      ; 0.65223568677902222E+00
18DF2683 E59CBCFE         02B8   286          .QUAD   ^X18DF2683E59CBCFE      ; -.17151232610031867E-14
242AD205 8E914060         02C0   287          .QUAD   ^X242AD2058E914060      ; 0.33113825085173019E+02
                          02C8   288  ;  Entry  9
00004000 C0E2400B         02C8   289          .QUAD   ^X00004000C0E2400B      ; 0.86729538440704346E+00
8CF25946 F483BCE9         02D0   290          .QUAD   ^X8CF25946F483BCE9      ; -.72039955175148569E-15
F491626E 77AC4064         02D8   291          .QUAD   ^XF491626E77AC4064      ; 0.40934948257615481E+02
                          02E0   292  ;  Entry 10
00000000 B97B4012         02E0   293          .QUAD   ^X00000000B97B4012      ; 0.11702833175659180E+01
F9F4D8EA 85FC3D00         02E8   294          .QUAD   ^XF9F4D8EA85FC3D00      ; 0.18344647350239910E-14
4E4578BA BE3F4068         02F0   295          .QUAD   ^X4E4578BABE3F4068      ; 0.49486311999291992E+02
```

```
                     02F8   296  ;  Entry 11
00006000 51BD401A    02F8   297          .QUAD   ^X0000600051BD401A      ; 0.16449599926986694E+01
81B4D2B1 A61D3CE3    0300   298          .QUAD   ^X81B4D2B1A61D3CE3      ; 0.54536632330091105E-15
77EDB336 5A15406D    0308   299          .QUAD   ^X77EDB3365A15406D      ; 0.58703787232967308E+02
                     0310   300  ;  Entry 12
0000A000 91004024    0310   301          .QUAD   ^X0000A00091004024      ; 0.25708019733428955E+01
5CBF9C63 DC35BD10    0318   302          .QUAD   ^X5CBF9C63DC35BD10      ; -.37437149019224865E-14
13586E14 2FAA4071    0320   303          .QUAD   ^X13586E142FAA4071      ; 0.68744777221303025E+02
                     0328   304  ;  Entry 13
0000C000 6FA14035    0328   305          .QUAD   ^X0000C0006FA14035      ; 0.53590154647827148E+01
408B5861 16AABD18    0330   306          .QUAD   ^X408B586116AABD18      ; -.53487296285387485E-14
ECD78FEF DB864073    0338   307          .QUAD   ^XECD78FEFDB864073      ; 0.79430088028242025E+02
                     0340   308
                     0340   309  ;
                     0340   310  ;  Tables to be used in POLYG for computing GATAND:  GATANDTAB1 is obtained
                     0340   311  ;  by multiplying the coefficients given in Hart et. al. (No. 4904) by
                     0340   312  ;  180/pi.  GATANDTAB2 is the same as GATANDTAB1 except that C0 is set to
                     0340   313  ;  180/pi - 64 instead of 180/pi.
                     0340   314  ;
                     0340   315
                     0340   316  GATANDTAB1:
96450669 1EC04031    0340   317          .QUAD   ^X964506691EC04031      ; C6 = 0.42800293924279389E+01
A4E1D5AC D3FCC034    0348   318          .QUAD   ^XA4E1D5ACD3FCC034      ; C5 = -.52070191752074786E+01
5E899E4D 76F94039    0350   319          .QUAD   ^X5E899E4D76F94039      ; C4 = 0.63661865935060939E+01
2808E93E 5EC6C040    0358   320          .QUAD   ^X2808E93E5EC6C040      ; C3 = -.81851113212942579E+01
7BA77B82 EB164046    0360   321          .QUAD   ^X7BA77B82EB164046      ; C2 = 0.11459155902555564E+02
2BF066ED 193DC053    0368   322          .QUAD   ^X2BF066ED193DC053      ; C1 = -.19098593171027403E+02
C1F81A63 A5DC406C    0370   323          .QUAD   ^XC1F81A63A5DC406C      ; C0 = 0.57295779513082323E+02
         00000007    0378   324  GATANDLEN1 = .- GATANDTAB1/8
                     0378   325
                     0378   326  GATANDTAB2:
96450669 1EC04031    0378   327          .QUAD   ^X964506691EC04031      ; C6 = 0.42800293924279389E+01
A4E1D5AC D3FCC034    0380   328          .QUAD   ^XA4E1D5ACD3FCC034      ; C5 = -.52070191752074786E+01
5E899E4D 76F94039    0388   329          .QUAD   ^X5E899E4D76F94039      ; C4 = 0.63661865935060939E+01
2808E93E 5EC6C040    0390   330          .QUAD   ^X2808E93E5EC6C040      ; C3 = -.81851113212942579E+01
7BA77B82 EB164046    0398   331          .QUAD   ^X7BA77B82EB164046      ; C2 = 0.11459155902555564E+02
2BF066ED 193DC053    03A0   332          .QUAD   ^X2BF066ED193DC053      ; C1 = -.19098593171027403E+02
                     03A8   333  G_PI_OV_180_M_64:
F0422CE1 D11FC03A    03A8   334          .QUAD   ^XF0422CE1D11FC03A      ; C0 = -.67042204869176789E+01
         00000007    03B0   335  GATANDLEN2 = .- GATANDTAB2/8
                     03B0   336
                     03B0   337
                     03B0   338  G_90:
00000000 80004076    03B0   339          .QUAD   ^X0000000080004076      ; 90.
                     03B8   340  G_M90:
00000000 8000C076    03B8   341          .QUAD   ^X000000008000C076      ; -90.
                     03C0   342  G_180:
00000000 80004086    03C0   343          .QUAD   ^X0000000080004086      ; 180
```

```
                        .SBTTL  MTHSGATAN - Standard G Floating Arc Tangent

;++
; FUNCTIONAL DESCRIPTION:
;
; GATAN - G floating point function
;
; GATAN is computed using the following steps:
;
;       1. If X > 11 then
;          a. Let W = 1/X.
;          b. Compute GATAN(W) = W*P(W**2), where P is a polynomial of
;             degree 6.
;          c. Set GATAN(X) = pi/2 - GATAN(W)
;       2. If 3/32 =< X =< 11 then
;          a. Obtain XHI by table look-up.
;          b. Compute Z = (X - XHI)/(1 + X*XHI).
;          c. Compute GATAN(Z) = Z*P(Z**2), where P is a polynomial of
;             degree 6.
;          d. Obtain GATAN(XHI) by table look-up.  GATAN(XHI) will have
;             two parts - the high order bits, GATAN_XHI_HI, and the low
;             order bits, GATAN_XHI_LO.
;          e. Compute GATAN(X) = GATAN_XHI_HI + (GATAN_XHI_LO + GATAN(Z)).
;       3. If 0 =< X < 3/32 then
;          a. Compute GATAN(X) = X + X*Q(X**2), where Q is a polynomial
;             of degree 6.
;       4. If X < 0 then
;          a. Compute Y = GATAN(|X|) using steps 1 to 3.
;          b. Set GATAN(X) = -Y.
;
;
; CALLING SEQUENCE:
;
;       Arctangent.wg.v = MTH$GATAN(x.rg.r)
;
; INPUT PARAMETERS:
;
;       LONG = 4                        ; define longword multiplier
;       x = 1 * LONG                    ; x is an angle in radians
;
; IMPLICIT INPUTS:      none
;
; OUTPUT PARAMETERS:
;
;       VALUE:  G floating arctangent angle of the argument
;
; IMPLICIT OUTPUTS:     none
;
; SIDE EFFECTS:
;
; Signals:      none
;
; NOTE: This procedure disables floating point underflow, enable integer
; overflow, causes no floating overflow or other arithmetic traps, and
; preserves enables across the call.
;
```

```
                        03C8   402 ;---
                        03C8   403
                        03C8   404
                 40FC   03C8   405          .ENTRY  MTH$GATAN, ACMASK          ; standard call-by-reference entry
                        03CA   406                                             ; disable DV (and FU), enable IV
                        03CA   407          MTH$FLAG_JACKET                    ; flag that this is a jacket procedure in
                        03CA                                                   ;
      6D  00000000'GF    9E   03CA          MOVAB   G^MTH$$JACKET_HND, (FP)
                        03D1                                                   ; set handler address to jacket
                        03D1                                                   ; handler
                        03D1
                        03D1   408                                             ; case of an error in special JSB routine
      50  04 BC 50FD    03D1   409          MOVG    @x(AP), R0                 ; R0/R1 = arg
            6F   10     03D6   410          BSBB    MTH$GATAN_R7               ; call special GATAN routine
                 04     03D8   411          RET                                ; return - result in R0
                        03D9   412
```

```
                                03D9   414          .SBTTL  MTH$GATAN2 - Standard G Floating Arctangent With 2 Arguments
                                03D9   415 ;++
                                03D9   416 ; FUNCTIONAL DESCRIPTION:
                                03D9   417 ;
                                03D9   418 ; GATAN2  - G floating point function
                                03D9   419 ;
                                03D9   420 ; GATAN2(X,Y) is computed as following:
                                03D9   421 ;
                                03D9   422 ;       If Y = 0 or X/Y > 2**57, GATAN2(X,Y) = PI/2 * (sign X)
                                03D9   423 ;       If Y > 0 and X/Y =< 2**57, GATAN2(X,Y) = GATAN(X/Y)
                                03D9   424 ;       If Y < 0 and X/Y =< 2**57, GATAN2(X,Y) = PI * (sign X) + GATAN(X/Y)
                                03D9   425 ;
                                03D9   426 ;
                                03D9   427 ; CALLING SEQUENCE:
                                03D9   428 ;
                                03D9   429 ;       Arctangent2.wg.v = MTH$GATAN2(x.rg.r, y.rg.r)
                                03D9   430 ;
                                03D9   431 ; INPUT PARAMETERS:
                                03D9   432 ;
                    00000004    03D9   433 ;       x = 1 * LONG                    ; x is the first argument
                    00000008    03D9   434 ;       y = 2 * LONG                    ; y is the second argument
                                03D9   435 ;
                                03D9   436 ; SIDE EFFECTS: See description of MTH$GATAN
                                03D9   437 ;
                                03D9   438 ;--
                                03D9   439
                                03D9   440
                        40FC    03D9   441          .ENTRY  MTH$GATAN2, ACMASK      ; standard call-by-reference entry
                                03DB   442                                          ; disable DV (and FU), enable IV
                                03DB   443          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure in
                                03DB                                                ;
    6D   00000000'GF    9E      03DB               MOVAB   G^MTH$$JACKET_HND, (FP)
                                03E2                                                ; set handler address to jacket
                                03E2                                                ; handler
                                03E2
                                03E2   444                                          ; case of an error in special JSB routine
      50   04 BC 50FD           03E2   445          MOVG    @x(AP), R0              ; R0/R1 = arg1
      52   08 BC 50FD           03E7   446          MOVG    @y(AP), R2              ; R2/R3 = arg2
                                03EC   447 ;
                                03EC   448 ; Test if Y = 0 or X/Y > 2**57
                                03EC   449 ;
                          33    13      03EC   450          BEQL    INF             ; branch to INF if Y = 0
54    50    800F 8F     AB      03EE   451          BICW3   #^X800F, R0, R4         ; R4 = exponent(X)
55    52    800F 8F     AB      03F4   452          BICW3   #^X800F, R2, R5         ; R5 = exponent(Y)
            54    55    A2      03FA   453          SUBW    R5, R4                  ; R4 = exponent(X) - exponent(Y)
      03A0 8F     54    B1      03FD   454          CMPW    R4, #58*16              ; compare R4 with 58
                    1D  14      0402   455          BGTR    INF                     ; if X/Y > 2**57, branch to INF
                                0404   456 ;
                                0404   457 ; Test if Y > 0 or Y < 0
                                0404   458 ;
                          52    B5      0404   459          TSTW    R2              ; test the sign of Y
                          16    14      0406   460          BGTR    A2PLUS          ; branch to A2PLUS if Y > 0
                          50    B5      0408   461          TSTW    R0              ; test the sign of X
                          09    18      040A   462          BGEQ    A1PLUS          ; branch to A1PLUS if X >= 0
                                040C   463 ;
                                040C   464 ; Y < 0 and X < 0 and X/Y =< 2**57
                                040C   465 ;
```

```
                35   10   040C   466          BSBB    MTH$GATAN_R7D           ; R0/R1 = GATAN(X/Y)
          50    FDB5 CF   42FD   040E   467          SUBG2   G_PI, R0            ; R0/R1 = -PI + GATAN(X/Y)
                     04   0414   468          RET                             ; return
                          0415   469 ;
                          0415   470 ; Y < 0 and X > 0 and X/Y =< 2**57
                          0415   471 ;
                          0415   472 A1PLUS:
                2C   10   0415   473          BSBB    MTH$GATAN_R7D           ; R0/R1 = GATAN(X/Y)
          50    FDAC CF   40FD   0417   474          ADDG2   G_PI, R0            ; R0/R1 = PI + GATAN(X/Y)
                     04   041D   475          RET                             ; return
                          041E   476 ;
                          041E   477 ; Y > 0 and X/Y =< 2**57
                          041E   478 ;
                          041E   479 A2PLUS:
                23   10   041E   480          BSBB    MTH$GATAN_R7D           ; R0/R1 = GATAN(X/Y)
                     04   0420   481          RET                             ; return
                          0421   482 ;
                          0421   483 ; Y = 0 or X/Y > 2**57
                          0421   484 ;
                          0421   485 INF:
                50   B5   0421   486          TSTW    R0                      ; test the sign of X
                08   14   0423   487          BGTR    1$                      ; branch if X > 0
                0C   13   0425   488          BEQL    2$                      ; branch if X = 0
          50    FDAD CF   7D     0427   489          MOVQ    G_MPI_OVER_2, R0        ; R0/R1 = GATAN(X/Y) = -PI/2
                     04   042C   490          RET                             ; return
                          042D   491
          50    FD9F CF   7D     042D   492 1$:       MOVQ    G_PI_OVER_2, R0         ; R0/R1 = GATAN(X/Y) = PI/2
                     04   0432   493          RET                             ; return
                          0433   494
                          0433   495 ;+
                          0433   496 ; Here if both X = 0 and Y = 0.  Signal INVALID ARG TO MATH LIBRARY
                          0433   497 ;-
                          0433   498
          50    01   0F   79   0433   499 2$:       ASHQ    #15, #1, R0             ; R0/R1 = reserved operand, copied
                          0437   500                                             ; to CHF$_MCH_SAVR0/R1 so handlers
                          0437   501                                             ; can change if they want to continue.
          7E    00'8F    9A   0437   502          MOVZBL  #MTH$K_INVARGMAT, -(SP) ; code for INVALID ARGMAT TO MATH LIBRARY
00000000'GF     01   FB   043B   503          CALLS   #1, G^MTH$$SIGNAL       ; Signal SEVERE error
                     04   0442   504          RET                             ; return if a handler says SS$_CONTINUE
```

```
                                0443   506          .SBTTL  MTH$GATAN_R7 - Special GATAN routine
                                0443   507
                                0443   508 ; Special GATAN - used by the standard routine, and directly.
                                0443   509 ;
                                0443   510 ; CALLING SEQUENCES:
                                0443   511 ;       save anything needed in R0:R7
                                0443   512 ;       MOVG    ...,R0                  ; input in R0/R1
                                0443   513 ;       JSB     MTH$GATAN_R7
                                0443   514 ;       return with result in R0/R1
                                0443   515 ; Note: This routine is written to avoid causing any integer overflows, floating
                                0443   516 ; overflows, or floating underflows or divide by 0 conditions, whether enabled or
                                0443   517 ; not.
                                0443   518 ;
                                0443   519 ; REGISTERS USED:
                                0443   520 ;       R0/R1 - Floating argument then result
                                0443   521 ;       R0:R5 - POLYG
                                0443   522 ;       R6/R7 - Y Guring POLYG
                                0443   523 ;
                                0443   524 ;
                                0443   525 ;
                                0443   526 MTH$GATAN_R7D:                          ; for local use only!
                     50   52 46FD  0443   527          DIVG2   R2, R0
                                0447   528 MTH$GATAN_R7::                         ; Special GATAN routine
                          50 53FD  0447   529          TSTG    R0                 ; R6 = X = argument
                          03   18  044A   530          BGEQ    POS_ARG
                        0089   31  044C   531          BRW     NEG_ARG            ; Branch to negative argument logic
                                044F   532 ;
                                044F   533 ; Argument is positive
                                044F   534 ;
          56    50    3FD8 8F   A3  044F   535 POS_ARG:SUBW3   #^X3FD8, R0, R6    ; Argument is less than 3/32,
                          5A   19  0455   536          BLSS    SMALL              ;  branch to small argument logic
                     56   006D 8F   B1  0457   537          CMPW    #^X006D, R6        ; Argument is greater that 11,
                          56   19  045C   538          BLSS    LARGE_ARG          ;  branch to large argument logic
                                045E   539 ;
                                045E   540 ; Logic for positive medium sized arguments.  Get pointer into GATAN_TABLE.
                                045E   541 ;
           56    56     FF 8F    9C  045E   542          ROTL    #-1, R6, R6        ; R6 = index into MTH$$AB_ATAN table
          56    FFFFFF00 8F    CA  0463   543          BICL    #-256, R6          ; zero high order bits of index
          53    00000000'GF    DE  046A   544          MOVAL   G^MTH$$AB_ATAN_V, R3   ; R3 = address of RTL vector entry
          53    00000000'GF    C0  0471   545          ADDL    G^MTH$$AB_ATAN_V, R3   ; R3 = address of MTH$$AB_ATAN table
                56    6346   90  0478   546          MOVB    (R3)[R6], R6       ; R6 = offset into GATAN_TABLE
            56    FB87 CF46   7E  047C   547          MOVAQ   GATAN_TABLE[R6], R6    ; R6 = pointer to XHI
                                0482   548 ;
                                0482   549 ; Compute Z
                                0482   550 ;
                     52    86   7D  0482   551          MOVQ    (R6)+, R2          ; R2 = XHI
                54   50    52 45FD  0485   552          MULG3   R2, R0, R4         ; R4 = X*XHI
                     54    08 40FD  048A   553          ADDG2   #1, R4             ; R4 = 1 + X*XHI
                     50    52 42FD  048E   554          SUBG2   R2, R0             ; R0 = X - XHI
                     50    54 46FD  0492   555          DIVG2   R4, R0             ; R0 = Z = (X - XHI)/(1 + X*XHI)
                                0496   556 ;
                                0496   557 ; Evaluate Z*P(Z**2)
                                0496   558 ;
                     7E    50   7D  0496   559          MOVQ    R0, -(SP)          ; Push Z onto the stack
                     50    50 44FD  0499   560          MULG2   R0, R0             ; R0 = Z**2
        FCB4 CF      06    50 55FD  049D   561          POLYG   R0, #GATANLEN1-1, GATANTAB1
                                04A4   562                                         ; R0 = P(Z**2)
```

```
                50   8E 44FD   04A4   563          MULG2   (SP)+, R0                ; R0 = GATAN(Z) = Z*P(Z**2)
                50   86 40FD   04A8   564          ADDG2   (R6)+, R0                ; R0 = GATAN_XHI_LO + GATAN(Z)
                50   66 40FD   04AC   565          ADDG2   (R6), R0                 ; R0 = GATAN(X) = GATAN_XHI_HI +
                                04B0   566                                           ;      (GATAN_XHI_LO + GATAN(Z))
                          05    04B0   567          RSB                              ; Return
                                04B1   568
                                04B1   569
                    00AB   31   04B1   570 SMALL:   BRW     SMALL_ARG               ; Dummy label used to avoid adding
                                04B4   571                                           ;   an extra insrtuction in the
                                04B4   572                                           ;   medium argument logic
                                04B4   573 ;
                                04B4   574 ; Large positive argument logic.
                                04B4   575 ;
                                04B4   576
                                04B4   577 LARGE_ARG:
     56   FB46 CF   50 47FD   04B4   578          DIVG3   R0, G_M1.0, R6           ; R6 = -W = -1/X
          50   56   56 45FD   04BB   579          MULG3   R6, R6, R0               ; R0 = W**2
FC91 CF   06   50 55FD   04C0   580          POLYG   R0, #GATANLEN1-1, GATANTAB1
                                04C7   581                                           ; R0 = P(W**2)
          50   56   56 44FD   04C7   582          MULG2   R6, R0                   ; R0 = GATAN(W) = -W*P(W**2)
     50   FD18 CF   50 40FD   04CB   583          ADDG2   G_PI_OVER_2_LO, R0       ;
     50   FD0A CF   50 40FD   04D1   584          ADDG2   G_PI_OVER_2_HI, R0       ; R0 = GATAN(X) = PI/2 - GATAN(W)
                     05    04D7   585          RSB                              ; Return
                                04D8   586
                                04D8   587 ;
                                04D8   588 ; Logic for negative arguments
                                04D8   589 ;
                                04D8   590
                                04D8   591 NEG_ARG:
56   50   BFD8 8F   A3   04D8   592          SUBW3   #^XBFD8, R0, R6          ; Argument is less than 3/32,
                7F   19   04DE   593          BLSS    SMALL_ARG                ;  branch to small argument logic
     56   006D 8F   B1   04E0   594          CMPW    #^X006D, R6              ; Argument is greater than 11,
                54   19   04E5   595          BLSS    N_LARGE_ARG              ;  branch to large argument logic
                                04E7   596 ;
                                04E7   597 ; Logic for negative medium sized arguments.  Get index into GATAN_TABLE.
                                04E7   598 ;
 56   56    FF 8F    9C   04E7   599          ROTL    #-1, R6, R6              ; R6 = index into MTH$$AB_ATAN table
56   FFFFFF00 8F    CA   04EC   600          BICL    #-256, R6                ; clear high order (unused) bits of ind
53   00000000'GF    DE   04F3   601          MOVAL   G^MTH$$AB_ATAN_V, R3     ; R3 = address of RTL vector entry
53   00000000'GF    C0   04FA   602          ADDL    G^MTH$$AB_ATAN_V, R3     ; R3 = address of MTH$$AB_ATAN table
     56    6346    90   0501   603          MOVB    (R3)[R6], R6             ; R6 = offset into GATAN_TABLE
  56   FAFE CF46    7E   0505   604          MOVAQ   GATAN_TABLE[R6], R6      ; R6 = pointer to XHI
                                050B   605 ;
                                050B   606 ; Compute Z
                                050B   607 ;
          52   86    7D   050B   608          MOVQ    (R6)+, R2                ; R2 = XHI
     54   50   52 45FD   050E   609          MULG3   R2, R0, R4               ; R4 = X*XHI
     54   08   54 43FD   0513   610          SUBG3   R4, #1, R4               ; R4 = 1 - X*XHI = 1 + X*(-XHI)
          50   52 40FD   0518   611          ADDG2   R2, R0                   ; R0 = X + XHI = X - (-XHI)
          50   54 46FD   051C   612          DIVG2   R4, R0                   ; R0 = Z
                                0520   613 ;
                                0520   614 ; Evaluate Z*P(Z**2)
                                0520   615 ;
          7E   50    7D   0520   616          MOVQ    R0, -(SP)                ; Push Z onto the stack
          50   50 44FD   0523   617          MULG2   R0, R0                   ; R0 = Z**2
FC2A CF   06   50 55FD   0527   618          POLYG   R0, #GATANLEN1-1, GATANTAB1
                                052E   619                                           ; R0 = P(Z**2)
```

```
                50   8E 44FD   052E   620          MULG2   (SP)+, R0                ; R0 = GATAN(Z) = Z*P(Z**2)
                50   86 42FD   0532   621          SUBG2   (R6)+, R0                ; R0 = GATAN_XHI_LO + GATAN(Z)
                50   66 42FD   0536   622          SUBG2   (R6), R0                 ; R0 = GATAN(X) = GATAN_XHI_HI +
                                053A   623                                           ;     (GATAN_XHI_LO + GATAN(Z))
                          05    053A   624          RSB                              ; Return
                                053B   625 ;
                                053B   626 ; Logic for large negative  arguments
                                053B   627 ;
                                053B   628
                                053B   629 N_LARGE_ARG:
    56   FABF   CF   50 47FD   053B   630          DIVG3   R0, G_M1.0, R6           ; R6 = W = 1/|X|
         50     56   56 45FD   0542   631          MULG3   R6, R6, R0               ; R0 = W**2
    FC0A CF     06   50 55FD   0547   632          POLYG   R0, #GATANLEN1-1, GATANTAB1
                                054E   633                                           ; R0 = P(W**2)
                50   56 44FD   054E   634          MULG2   R6, R0                   ; R0 = GATAN(W) = W*P(W**2)
         50     FC91 CF 42FD   0552   635          SUBG2   G_PI_OVER_2_LO, R0       ;
         50     FC83 CF 42FD   0558   636          SUBG2   G_PI_OVER_2_HI, R0       ; R0 = GATAN(X) = GATAN(W) - PI/2
                          05    055E   637          RSB                              ; Return
                                055F   638
                                055F   639 ;
                                055F   640 ; Small argument logic.
                                055F   641 ;
                                055F   642
                                055F   643 SMALL_ARG:
                56   50   7D   055F   644          MOVQ    R0, R6                   ; R6 = argument = X
         50     8000 8F   AA   0562   645          BICW    #^X8000, R0              ; R0 = |X|
         50     3E70 8F   B1   0567   646          CMPW    #^X3E70, R0              ; Compare 2^-26 to |X|
                     04   19   056C   647          BLSS    1$                       ; Needs polynomial evaluation
                50   56   7D   056E   648          MOVQ    R6, R0                   ; Return with answer equal to
                          05    0571   649          RSB                              ;    argument
                                0572   650
                50   50 44FD   0572   651 1$:      MULG2   R0, R0                   ; R0 = X**2
    FC13 CF     06   50 55FD   0576   652          POLYG   R0, #GATANLEN2-1, GATANTAB2
                                057D   653                                           ; R0 = Q(X**2)
                50   56 44FD   057D   654          MULG2   R6, R0                   ; R0 = X*Q(X**2)
                50   56 40FD   0581   655          ADDG2   R6, R0                   ; R0 = GATAN(X) = X + X*Q(X**2)
                          05    0585   656          RSB                              ; Return
                                0586   657
```

```
                        0586   659          .SBTTL  MTHSGATAND - Standard G Floating Arc Tangent
                        0586   660
                        0586   661
                        0586   662 ;++
                        0586   663 ; FUNCTIONAL DESCRIPTION:
                        0586   664 ;
                        0586   665 ; GATAND - G floating point function
                        0586   666 ;
                        0586   667 ; GATAND is computed using the following steps:
                        0586   668 ;
                        0586   669 ;     1. If X > 11 then
                        0586   670 ;        a. Let W = 1/X.
                        0586   671 ;        b. Compute GATAND(W) = W*P(W**2), where P is a polynomial of
                        0586   672 ;           degree 6.
                        0586   673 ;        c. Set GATAND(X) = pi/2 - GATAND(W)
                        0586   674 ;     2. If 3/32 =< X =< 11 then
                        0586   675 ;        a. Obtain XHI by table look-up.
                        0586   676 ;        b. Compute Z = (X - XHI)/(1 + X*XHI).
                        0586   677 ;        c. Compute GATAND(Z) = Z*P(Z**2), where P is a polynomial of
                        0586   678 ;           degree 6.
                        0586   679 ;        d. Obtain GATAND(XHI) by table look-up.  GATAND(XHI) will have
                        0586   680 ;           two parts - the high order bits, GATAND_XHI_HI, and the low
                        0586   681 ;           order bits, GATAND_XHI_LO.
                        0586   682 ;        e. Compute GATAND(X) = GATAND_XHI_HI + (GATAND_XHI_LO + GATAND(Z)).
                        0586   683 ;     3. If 0 =< X < 3/32 then
                        0586   684 ;        a. Compute GATAND(X) = X + X*Q(X**2), where Q is a polynomial
                        0586   685 ;           of degree 6.
                        0586   686 ;     4. If X < 0 then
                        0586   687 ;        a. Compute Y = GATAND(!X!) using steps 1 to 3.
                        0586   688 ;        b. Set GATAND(X) = -Y.
                        0586   689 ;
                        0586   690 ;
                        0586   691 ; CALLING SEQUENCE:
                        0586   692 ;
                        0586   693 ;       Arctangent.wg.v = MTH$GATAND(x.rg.r)
                        0586   694 ;
                        0586   695 ; INPUT PARAMETERS:
                        0586   696 ;
00000004                0586   697 ;       LONG = 4                        ; define longword multiplier
00000004                0586   698 ;       x = 1 * LONG                    ; x is an angle in degrees
                        0586   699 ;
                        0586   700 ; IMPLICIT INPUTS:       none
                        0586   701 ;
                        0586   702 ; OUTPUT PARAMETERS:
                        0586   703 ;
                        0586   704 ;       VALUE:  G floating arctangent angle of the argument
                        0586   705 ;
                        0586   706 ; IMPLICIT OUTPUTS:      none
                        0586   707 ;
                        0586   708 ; SIDE EFFECTS:
                        0586   709 ;
                        0586   710 ; Signals:       none
                        0586   711 ;
                        0586   712 ; NOTE: This procedure disables floating point underflow, enable integer
                        0586   713 ; overflow, causes no floating overflow or other arithmetic traps, and
                        0586   714 ; preserves enables across the call.
                        0586   715 ;
```

```
                        0586   716 ;---
                        0586   717
                        0586   718
                 40FC   0586   719          .ENTRY  MTHSGATAND, ACMASK      ; standard call-by-reference entry
                        0588   720                                          ; disable DV (and FU), enable IV
                        0588   721          MTHSFLAG_JACKET                 ; flag that this is a jacket procedure in
                        0588                                                ;
6D  00000000'GF   9E    0588               MOVAB   G^MTHS$JACKET_HND, (FP)
                        058F                                                ; set handler address to jacket
                        058F                                                ; handler
                        058F
                        058F   722                                          ; case of an error in special JSB routine
 50  04 BC 50FD    8F    058F   723          MOVG    @x(AP), R0              ; R0/R1 = arg
               6F    10   0594   724          BSBB    MTHSGATAND_R7           ; call special GATAND routine
                     04   0596   725          RET                             ; return - result in R0
                        0597   726
```

```
                                0597   728          .SBTTL  MTH$GATAND2 - Standard G Floating Arctangent With 2 Arguments
                                0597   729 ;++
                                0597   730 ; FUNCTIONAL DESCRIPTION:
                                0597   731 ;
                                0597   732 ; GATAND2  - G floating point function
                                0597   733 ;
                                0597   734 ; GATAND2(X,Y) is computed as following:
                                0597   735 ;
                                0597   736 ;       If Y = 0 or X/Y > 2**57, GATAND2(X,Y) = 90 * (sign X)
                                0597   737 ;       If Y > 0 and X/Y =< 2**57, GATAND2(X,Y) = GATAND(X/Y)
                                0597   738 ;       If Y < 0 and X/Y =< 2**57, GATAND2(X,Y) = 180 * (sign X) + GATAND(X/Y)
                                0597   739 ;
                                0597   740 ;
                                0597   741 ; CALLING SEQUENCE:
                                0597   742 ;
                                0597   743 ;       Arctangent2.wg.v = MTH$GATAND2(x.rg.r, y.rg.r)
                                0597   744 ;
                                0597   745 ; INPUT PARAMETERS:
                                0597   746
                       00000004 0597   747          x = 1 * LONG                    ; x is the first argument
                       00000008 0597   748          y = 2 * LONG                    ; y is the second argument
                                0597   749
                                0597   750 ; SIDE EFFECTS: See description of MTH$GATAND
                                0597   751 ;
                                0597   752 ;--
                                0597   753
                                0597   754
                         40FC   0597   755          .ENTRY  MTH$GATAND2, ACMASK     ; standard call-by-reference entry
                                0599   756                                          ; disable DV (and FU), enable IV
                                0599   757          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure in
                                0599
          6D   00000000'GF      9E   0599                   MOVAB   G^MTH$$JACKET_HND, (FP)
                                05A0                                               ; set handler address to jacket
                                05A0                                               ; handler
                                05A0
                                05A0   758                                         ; case of an error in special JSB routine
           50   04 BC 50FD      05A0   759          MOVG    @x(AP), R0              ; R0/R1 = arg1
           52   08 BC 50FD      05A5   760          MOVG    @y(AP), R2              ; R2/R3 = arg2
                                05AA   761 ;
                                05AA   762 ; Test if Y = 0 or X/Y > 2**57
                                05AA   763 ;
                          53    13   05AA   764          BEQL    INF_DEG             ; branch to INF_DEG if Y = 0
     54   50    800F   BF    AB   05AC   765          BICW3   #^X800F, R0, R4         ; R4 = exponent(X)
     55   52    800F   BF    AB   05B2   766          BICW3   #^X800F, R2, R5         ; R5 = exponent(Y)
               54     55    A2   05B8   767          SUBW    R5, R4                  ; R4 = exponent(X) - exponent(Y)
          03A0 8F     54    B1   05BB   768          CMPW    R4, #58*16              ; compare R4 with 58
                      1D    14   05C0   769          BGTR    INF_DEG                 ; if X/Y > 2**57, branch to INF_DEG
                                05C2   770 ;
                                05C2   771 ; Test if Y > 0 or Y < 0
                                05C2   772 ;
                          52    B5   05C2   773          TSTW    R2                  ; test the sign of Y
                          16    14   05C4   774          BGTR    A2PLUSD             ; branch to A2PLUSD if Y > 0
                          50    B5   05C6   775          TSTW    R0                  ; test the sign of X
                          09    18   05C8   776          BGEQ    A1PLUSD             ; branch to A1PLUSD if X >= 0
                                05CA   777 ;
                                05CA   778 ; Y < 0 and X < 0 and X/Y =< 2**57
                                05CA   779 ;
```

```
                35   10   05CA   780                BSBB    MTH$GATAND_R7D          ; R0/R1 = GATAND(X/Y)
          50    FDEF CF   42FD   05CC   781                SUBG2   G_180, R0           ; R0/R1 = -180 + GATAND(X/Y)
                     04   05D2   782                RET                             ; return
                          05D3   783 ;
                          05D3   784 ; Y < 0 and X > 0 and X/Y =< 2**57
                          05D3   785 ;
                          05D3   786 A1PLUSD:
                2C   10   05D3   787                BSBB    MTH$GATAND_R7D          ; R0/R1 = GATAND(X/Y)
          50    FDE6 CF   40FD   05D5   788                ADDG2   G_180, R0           ; R0/R1 = 180 + GATAND(X/Y)
                     04   05DB   789                RET                             ; return
                          05DC   790 ;
                          05DC   791 ; Y > 0 and X/Y =< 2**57
                          05DC   792 ;
                          05DC   793 A2PLUSD:
                23   10   05DC   794                BSBB    MTH$GATAND_R7D          ; R0/R1 = GATAND(X/Y)
                     04   05DE   795                RET                             ; return
                          05DF   796 ;
                          05DF   797 ; Y = 0 or X/Y > 2**57
                          05DF   798 ;
                          05DF   799 INF_DEG:
                50   B5   05DF   800                TSTW    R0                      ; test the sign of X
                08   14   05E1   801                BGTR    1$                      ; branch if X > 0
                0C   13   05E3   802                BEQL    2$                      ; branch if X = 0
          50    FDCF CF   7D   05E5   803                MOVQ    G_M90, R0       ; R0/R1 = GATAND(X/Y) = -90
                     04   05EA   804                RET                             ; return
                          05EB   805
          50    FDC1 CF   7D   05EB   806 1$:      MOVQ    G_90, R0                ; R0/R1 = GATAND(X/Y) = 90
                     04   05F0   807                RET                             ; return
                          05F1   808
                          05F1   809 ;+
                          05F1   810 ; Here if both X = 0 and Y = 0.  Signal INVALID ARG TO MATH LIBRARY
                          05F1   811 ;-
                          05F1   812
             50   01   0F   79   05F1   813 2$:      ASHQ    #15, #1, R0             ; R0/R1 = reserved operand, co180ed
                          05F5   814                                              ; to CHF$_MCH_SAVR0/R1 so handlers
                          05F5   815                                              ; can change if they want to continue.
                7E   00'8F   9A   05F5   816                MOVZBL  #MTH$K_INVARGMAT, -(SP) ; code for INVALID ARGMAT TO MATH LIBRARY
00000000'GF      01   FB   05F9   817                CALLS   #1, G^MTH$$SIGNAL       ; Signal SEVERE error
                     04   0600   818                RET                             ; return if a handler says SS$_CONTINUE
```

```
                                0601    820          .SBTTL  MTH$GATAND_R7 - Special GATAND routine
                                0601    821
                                0601    822 ; Special GATAND - used by the standard routine, and directly.
                                0601    823 ;
                                0601    824 ; CALLING SEQUENCES:
                                0601    825 ;       save anything needed in R0:R7
                                0601    826 ;       MOVG    ...,R0                  ; input in R0/R1
                                0601    827 ;       JSB     MTH$GATAND_R7
                                0601    828 ;       return with result in R0/R1
                                0601    829 ; Note: This routine is written to avoid causing any integer overflows, floating
                                0601    830 ; overflows, or floating underflows or divide by 0 conditions, whether enabled or
                                0601    831 ; not.
                                0601    832 ;
                                0601    833 ; REGISTERS USED:
                                0601    834 ;       R0/R1 - Floating argument then result
                                0601    835 ;       R0:R5 - POLYG
                                0601    836 ;       R6/R7 - Y during POLYG
                                0601    837
                                0601    838
                                0601    839
                                0601    840 MTH$GATAND_R7D:                         ; for local use only!
                     50   52 46FD   0601    841          DIVG2   R2, R0
                                0605    842 MTH$GATAND_R7::                         ; Special GATAND routine
                          50 53FD   0605    843          TSTG    R0                 ; R6 = X = argument
                          03   18   0608    844          BGEQ    POS_ARGD
                        0083   31   060A    845          BRW     NEG_ARGD           ; Branch to negative argument logic
                                060D    846 ;
                                060D    847 ; Argument is positive
                                060D    848 ;
                                060D    849 POS_ARGD:
          56   50   3FD8 8F    A3   060D    850          SUBW3   #^X3FD8, R0, R6    ; Argument is less than 3/32,
                               5A   19   0613    851          BLSS    SMALLD             ;  branch to small argument logic
               56   006D 8F    B1   0615    852          CMPW    #^X006D, R6        ; Argument is greater that 11,
                               56   19   061A    853          BLSS    LARGE_ARGD         ;  branch to large argument logic
                                061C    854 ;
                                061C    855 ; Logic for positive medium sized arguments.  Get pointer into GATAND_TABLE.
                                061C    856 ;
           56   56    FF 8F    9C   061C    857          ROTL    #-1, R6, R6        ; R6 = index into MTH$$AB_ATAN table
          56   FFFFFF00 8F    CA   0621    858          BICL    #-256, R6          ; zero high order bits of index
          53   00000000'GF    DE   0628    859          MOVAL   G^MTH$$AB_ATAN_V, R3 ; R3 = address of RTL vector entry
          53   00000000'GF    C0   062F    860          ADDL    G^MTH$$AB_ATAN_V, R3 ; R3 = address of MTH$$AB_ATAN table
            56     6346    90   0636    861          MOVB    (R3)[R6], R6       ; R6 = offset into GATAND_TABLE
            56   FBB1 CF46   7E   063A    862          MOVAQ   GATAND_TABLE[R6], R6 ; R6 = pointer to XHI
                                0640    863 ;
                                0640    864 ; Compute z
                                0640    865 ;
                          52   86   7D   0640    866          MOVQ    (R6)+, R2          ; R2 = XHI
                     54   50   52 45FD   0643    867          MULG3   R2, R0, R4         ; R4 = X*XHI
                          54   08 40FD   0648    868          ADDG2   #1, R4             ; R4 = 1 + X*XHI
                          50   52 42FD   064C    869          SUBG2   R2, R0             ; R0 = X - XHI
                          50   54 46FD   0650    870          DIVG2   R4, R0             ; R0 = Z = (X - XHI)/(1 + X*XHI)
                                0654    871 ;
                                0654    872 ; Evaluate Z*P(Z**2)
                                0654    873 ;
                          7E   50   7D   0654    874          MOVQ    R0, -(SP)          ; Push Z onto the stack
                          50   50 44FD   0657    875          MULG2   R0, R0             ; R0 = Z**2
        FCDE CF    06   50 55FD   065B    876          POLYG   R0, #GATANDLEN1-1, GATANDTAB1
```

```
                                0662   877
                                0662   878          MULG2   (SP)+, R0                      ; R0 = GATAND(Z) = Z*Q(Z**2)
                                0666   879          ADDG2   (R6)+, R0                      ; R0 = GATAND_XHI_LO + GATAND(Z)
                                066A   880          ADDG2   (R6), R0                       ; R0 = GATAND(X) = GATAND_XHI_HI +
                                066E   881                                                 ;      (GATAND_XHI_LO + GATAND(Z))
                                066E   882          RSB                                    ; Return
                                066F   883
                                066F   884
                           009F 31      066F   885  SMALLD: BRW     SMALL_ARGD             ; Dummy label used to avoid adding
                                0672   886                                                 ;   an extra insrtuction in the
                                0672   887                                                 ;   medium argument logic
                                0672   888  ;
                                0672   889  ; Large positive argument logic.
                                0672   890  ;
                                0672   891
                                0672   892  LARGE_ARGD:
56      F988 CF    50 47FD      0672   893          DIVG3   R0, G_M1.0, R6                  ; R6 = -W = -1/X
     50      56    56 45FD      0679   894          MULG3   R6, R6, R0                     ; R0 = W**2
FCBB CF      06    50 55FD      067E   895          POLYG   R0, #GATANDLEN1-1, GATANDTAB1
                                0685   896                                                 ; R0 = P(W**2)
        50   56    56 44FD      0685   897          MULG2   R6, R0                         ; R0 = -GATAND(Z) = -Z*P(W**2)
     50      FD22 CF    40FD    0689   898          ADDG2   G_90, R0                       ; R0 = GATAND(X) = 90 - GATAND(Z)
                        05      068F   899          RSB                                    ; Return
                                0690   900
                                0690   901  ;
                                0690   902  ; Logic for negative arguments
                                0690   903  ;
                                0690   904
                                0690   905  NEG_ARGD:
56   50      BFD8 8F    A3      0690   906          SUBW3   #^XBFD8, R0, R6                ; Argument is less than 3/32,
                        79  19  0696   907          BLSS    SMALL_ARGD                     ;  branch to small argument logic
     56      006D 8F    B1      0698   908          CMPW    #^X006D, R6                    ; Argument is greater than 11,
                        54  19  069D   909          BLSS    N_LARGE_ARGD                   ;  branch to large argument logic
                                069F   910  ;
                                069F   911  ; Logic for negative medium sized arguments.  Get index into GATAND_TABLE.
                                069F   912  ;
 56   56      FF 8F    9C      069F   913          ROTL    #-1, R6, R6                    ; R6 = index into MTH$$AB_ATAN table
56    FFFFFF00 8F    CA      06A4   914          BICL    #-256, R6                      ; clear high order (unused) bits of ind
53    00000000'GF    DE      06AB   915          MOVAL   G^MTH$$AB_ATAN_V, R3           ; R3 = address of RTL vector entry
53    00000000'GF    C0      06B2   916          ADDL    G^MTH$$AB_ATAN_V, R3           ; R3 = address of MTH$$AB_ATAN table
     56     6346    90      06B9   917          MOVB    (R3)[R6], R6                   ; R6 = offset into GATAND_TABLE
  56    FB2E CF46    7E      06BD   918          MOVAQ   GATAND_TABLE[R6], R6           ; R6 = pointer to XHI
                                06C3   919  ;
                                06C3   920  ; Compute Z
                                06C3   921  ;
        52    86     7D      06C3   922          MOVQ    (R6)+, R2                      ; R2 = XHI
     54 50    52 45FD         06C6   923          MULG3   R2, R0, R4                     ; R4 = X*XHI
     54 08    54 43FD         06CB   924          SUBG3   R4, #1, R4                     ; R4 = 1 - X*XHI = 1 + X*(-XHI)
        50    52 40FD         06D0   925          ADDG2   R2, R0                         ; R0 = X + XHI = X - (-XHI)
        50    54 46FD         06D4   926          DIVG2   R4, R0                         ; R0 = Z
                                06D8   927  ;
                                06D8   928  ; Evaluate Z*P(Z**2)
                                06D8   929  ;
        7E    50     7D      06D8   930          MOVQ    R0, -(SP)                      ; Push Z onto the stack
        50    50 44FD         06DB   931          MULG2   R0, R0                         ; R0 = Z**2
FC5A CF    06    50 55FD      06DF   932          POLYG   R0, #GATANDLEN1-1, GATANDTAB1
                                06E6   933                                                 ; R0 = P(Z**2)
```

```
                50   8E 44FD   06E6   934          MULG2   (SP)+, R0                ; R0 = GATAND(Z) = Z*P(Z**2)
                50   86 42FD   06EA   935          SUBG2   (R6)+, R0                ; R0 = GATAND_XHI_LO + GATAND(Z)
                50   66 42FD   06EE   936          SUBG2   (R6), R0                 ; R0 = GATAND(X) = GATAND_XHI_HI +
                                06F2   937                                           ;      (GATAND_XHI_LO + GATAND(Z))
                         05     06F2   938          RSB                              ; Return
                                06F3   939 ;
                                06F3   940 ; Logic for large negative  arguments
                                06F3   941 ;
                                06F3   942
                                06F3   943 N_LARGE_ARGD:
    56   F907 CF   50 47FD   06F3   944          DIVG3   R0, G_M1.0, R6           ; R6 = W = 1/!X!
         50   56   56 45FD   06FA   945          MULG3   R6, R6, R0               ; R0 = W**2
FC3A CF   06   50 55FD   06FF   946          POLYG   R0, #GATANDLEN1-1, GATANDTAB1
                         0706   947                                           ; R0 = P(W**2)
         50   56 44FD   0706   948          MULG2   R6, R0                   ; R0 = GATAND(W) = W*P(W**2)
    50   FCA1 CF 42FD   070A   949          SUBG2   G_90, R0                 ; R0 = GATAND(X) = GATAND(W) - 90
                         0710   950          RSB                              ; Return
                                0711   951
                                0711   952 ;
                                0711   953 ; Small argument logic.
                                0711   954 ;
                                0711   955
                                0711   956 SMALL_ARGD:
                56   50 50FD   0711   957          MOVG    R0, R6                   ; R6 = argument = X
                     2D   13   0715   958          BEQL    3$                       ;
         50   8000 8F   AA     0717   959          BICW    #^X8000, R0              ; R0 = !X!
         50   3E70 8F   B1     071C   960          CMPW    #^X3E70, R0              ; Compare 2^-26 to !X!
                     09   19   0721   961          BLSS    1$                       ; Needs polynomial evaluation
50   56   FC80 CF 45FD   0723   962          MULG3   G_PI_OV_180_M_64, R6, R0 ; R0 = X*(pi/180 - 64)
                     0F   11   072A   963          BRB     2$                       ;
                50   50 44FD   072C   964 1$:      MULG2   R0, R0                   ; R0 = X**2
FC41 CF   06   50 55FD   0730   965          POLYG   R0, #GATANDLEN2-1, GATANDTAB2
                         0737   966                                           ; R0 = Q(X**2)
                50   56 44FD   0737   967          MULG2   R6, R0                   ; R0 = X*Q(X**2)
         56   0060 8F   A0     073B   968 2$:      ADDW    #^X60, R6                ; R6 = X*2**6
                50   56 40FD   0740   969          ADDG2   R6, R0                   ; R0 = GATAND(X) = X*2**6 + X*Q(X**2)
                         05     0744   970 3$:      RSB                              ; Return
                                0745   971
                                0745   972
                                0745   973          .END
```

```
A1PLUS                          00000415 R      01
A1PLUSD                         000005D3 R      01
A2PLUS                          0000041E R      01
A2PLUSD                         000005DC R      01
ACMASK                        = 000040FC
GATANDLEN1                    = 00000007
GATANDLEN2                    = 00000007
GATANDTAB1                      00000340 R      01
GATANDTAB2                      00000378 R      01
GATAND_TABLE                    000001F0 R      01
GATANLEN1                     = 00000007
GATANLEN2                     = 00000007
GATANTAB1                       00000158 R      01
GATANTAB2                       00000190 R      01
GATAN_TABLE                     00000008 R      01
G_180                           000003C0 R      01
G_90                            000003B0 R      01
G_M1.0                          00000000 R      01
G_M90                           000003B8 R      01
G_MPI_OVER_2                    000001D8 R      01
G_PI                            000001C8 R      01
G_PI_OVER_2                     000001D0 R      01
G_PI_OVER_2_HI                  000001E0 R      01
G_PI_OVER_2_LO                  000001E8 R      01
G_PI_OV_180_M_64                000003A8 R      01
INF                             00000421 R      01
INF_DEG                         000005DF R      01
LARGE_ARG                       000004B4 R      01
LARGE_ARGD                      00000672 R      01
LONG                          = 00000004
MTH$$AB_ATAN_V                  ********    X   00
MTH$$JACKET_AND                 ********    X   01
MTH$$SIGNAL                     ********    X   00
MTH$GATAN                       000003C8 RG     01
MTH$GATAN2                      000003D9 RG     01
MTH$GATAND                      00000586 RG     01
MTH$GATAND2                     00000597 RG     01
MTH$GATAND_R7                   00000605 RG     01
MTH$GATAND_R7D                  00000601 R      01
MTH$GATAN_R7                    00000447 RG     01
MTH$GATAN_R7D                   00000443 R      01
MTH$K_INVARGMAT                 ********      X 00
NEG_ARG                         000004D8 R      01
NEG_ARGD                        00000690 R      01
N_LARGE_ARG                     0000053B R      01
N_LARGE_ARGD                    000006F3 R      01
POS_ARG                         0000044F R      01
POS_ARGD                        0000060D R      01
SMALL                           000004B1 R      01
SMALLD                          0000066F R      01
SMALL_ARG                       0000055F R      01
SMALL_ARGD                      00000711 R      01
X                             = 00000004
Y                             = 00000008
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_MTH$CODE	00000745 (1861.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.08	00:00:01.17
Command processing	114	00:00:00.63	00:00:03.80
Pass 1	120	00:00:02.54	00:00:07.51
Symbol table sort	0	00:00:00.03	00:00:00.03
Pass 2	180	00:00:02.14	00:00:08.38
Symbol table output	7	00:00:00.06	00:00:00.06
Psect synopsis output	3	00:00:00.03	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	460	00:00:05.51	00:00:20.98

The working set limit was 1050 pages.
16379 bytes (32 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 54 non-local and 8 local symbols.
1033 source lines were read in Pass 1, producing 24 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:MTHGATAN/OBJ=OBJ$:MTHGATAN MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC